Coded Caching with Heterogeneous User Profiles

Ciyuan Zhang, *Student Member, IEEE* Su Wang, Vaneet Aggarwal, *Senior Member, IEEE* and Borja Peleato, *Senior Member, IEEE*

***Abstract*—Coded caching utilizes pre-fetching during off-peak hours and multi-casting for delivery in order to balance the traffic load in communication networks. Several works have studied the achievable peak and average rates under different conditions: variable file lengths or popularities, variable cache sizes, decentralized networks, etc. However, very few have considered the possibility of heterogeneous user profiles, despite modern content providers are investing heavily in categorizing users according to their habits and preferences.**

This paper proposes three coded caching schemes with uncoded pre-fetching for scenarios where end users are grouped into classes with different file demand sets (FDS). One scheme ignores the difference between the classes, another ignores the similarities between them and the third decouples the delivery of files common to all FDS from those unique to a single class. The transmission rates of the three schemes are compared with a lower bound to evaluate their gap to optimality, and with each other to show that each scheme can outperform the other two when certain conditions are met.

***Index Terms*—Coded caching, User profiles, Content distribution networks, Peak and average rate.**

I. Introduction

The recent information explosion is constantly pushing the limits of communication networks, users always want more information at faster speeds and with minimal latency. Network operators hope to address this problem by pushing the content and computation closer to the end users, in what is commonly known as fog networking [2]. Having multiple caches distributed across the network helps balance the load over the internet backbone, but does not alleviate the congestion that often arises at the edge of the network during peak hours. Coded caching was introduced as a powerful solution for solving this problem and has since been used in a wide range of areas, such as industrial IoT [3], 5G wireless communications [4], [5], and medical data sharing [6].

A coded caching scheme consists of a placement and a delivery phase. The placement phase takes place during off-peak hours, when there are spare resources in the network. The server partitions all the files into segments and stores them in the users' caches. The delivery phase takes place during peak hours, when multiple (if not all) users have file requests. The server attempts to fulfill all those requests with minimal information transmitted, by leveraging the segments cached during the placement phase. It has long been known that proactively caching popular content during off-peak hours reduces the total information to be transmitted when that content is requested. This gain depends on the hit rates on the local cache of the end users, so it is known as local caching gain. However, Maddah-Ali and Niesen's seminal paper [7] recently showed that the overall transmission rate in point-to-multipoint links can be reduced further by carefully coordinating the cached segments and using a coded delivery scheme. This gain depends on the segments shared by the different user subgroups and is therefore known as global caching gain.

In [7], Maddah-Ali and Niesen proposed a coded caching scheme which maximizes multicasting opportunities for the worst case user demands. Subsequent works focused on lowering the peak rate in different scenarios [8], [9]. However, these papers adopted homogeneous models which do not fit most practical systems where coded caching could potentially be used. Some recent works have analyzed the transmission rate of coded caching systems with full heterogeneity: [10] considered different file sizes, cache sizes, and user dependent file popularity, but only for two users and two files. Centralized and decentralized coded caching schemes with heterogeneous user cache sizes were studied in [11] and [12], respectively, but they ignored the users' diverse preference over files. The work in [13] provided an optimization theoretic analysis of coded caching systems with various heterogeneities (cache size, file length, and file popularity) and demonstrated that Maddah-Ali and Niesen's original scheme from [7] is optimal for problems with uniform file size, popularity, and cache size. Unfortunately, the results in [13] are derived numerically, without theoretical evidence. Furthermore, it does not address user heterogeneity, which is the focus of this paper. A scenario with heterogeneous file popularities was addressed in [14] by partitioning files into groups such that, within each group, the files have approximately equal popularities. However, it again assumed that the popularity of each file was identical for every user. Additional research on coded caching has extended it to topics such as device-to-device caching [15], hierarchical caching [16], location-based content [17], and distinct file sizes [18].

Papers like [19]–[21] have addressed the significance of predicting users behavior according to their preferences. This mirrors the current trend of online video streaming companies like Hulu and Netflix which spend a considerable amount of resources investigating their customers' habits and categorizing them according to their streaming preferences. The paper [22] utilized game theory to analyze the transmission cost of a centralized coded caching system when the users present heterogeneous preferences over the files requested, but

This work was presented in part at ICC 2020 [1].

C. Zhang, S. Wang, and V. Aggarwal are with Purdue University, West Lafayette, IN 47907, USA. Email:{zhan3375,wang2506,vaneet}@purdue.edu

B. Peleato is with Carlos III University of Madrid, Leganes, Spain. Email: bpeleato@ing.uc3m.es

This work was funded in part by the Comunidad Autónoma de Madrid under Grant 2020-T1/TIC-20698, the CONEX-Plus Programme - Marie-Sklodowska Curie COFUND Action (H2020-MSCA-COFUND-2017- GA 801538) and Banco Santander, and the U.S. National Science Foundation under grants CNS-1618335 and CIF-2149588.

it neglected the alternative of grouping similar users together. The recent paper [23] categorized users into two groups: VIP and non-VIP. It proposed schemes so that the VIP group obtains better experience (lower transmission rate) than the non-VIP group. However, it only considered the decentralized coded caching system model and merely paid attention to specific users instead of the whole ensemble. Another coded caching scheme with user grouping was proposed in [24] for wireless channels, and its results indicate that grouping users based on their channel conditions is beneficial for reducing transmission time, especially for small cache sizes. Our prior conference paper [25] addressed a system where the users are grouped into classes with similar file interests. It proposed three coded caching schemes for this scenario and studied their peak rate, but it did not provide a comprehensive comparison between them. This paper will do that and study the subject in more detail.

Most existing works have focused on studying the peak rather than the average rate of coded caching systems. This is mainly due to the fact that the average rate is highly dependent on the distribution of the requests and that the peak rate is an important factor in the design of small networks. When the number of files and users is large, however, the peak rate is very rarely reached because it is common for different users to request same files. Hence, studying the average rate of transmission can be more useful towards modeling and developing practical caching schemes. There have been works studying the average rate, e.g., [26], [27], but they assumed the same distribution of requests for all users. The scenario with heterogeneous user profiles was thoroughly analyzed in [28], [29], but just for the case of two users. Our prior work [1] studied the average rate resulting from the three schemes proposed in [25] and compared their asymptotic performance.

The main contributions of this paper include: 1) characterizing the peak and average rates of the three schemes proposed in [25] for a coded caching system with heterogeneous user profiles; 2) deriving lower bounds for the peak rate of the three schemes; 3) proposing a cache distribution method which results in minimal peak and average rate for one of the schemes when the caches are relatively small compared with the size of the library; 4) comparing the peak transmission rate of the three schemes analytically to provide insights for deciding which scheme should be chosen given the system's parameters.

The paper will be organized as follows: Section II introduces our system model and the notation to be used throughout the paper. Section III describes the three coded caching schemes being proposed and analyzed. Section IV derives a lower bound for the peak rate of a coded caching scheme with heterogeneous user profiles and compares the peak rate of the three schemes with that bound. Section V studies how to optimally distribute the cache among the different types of files and compares the peak rate of the three schemes according to the cache size. Finally, Section VI provides numerical simulation results to illustrate and support our derivations, and Section VII concludes the paper.

II. Background

A. System Model

This paper considers a system with a single server storing N files of size F, which is connected through an error-free broadcast link to K end users equipped with cache memories of size MF each. The K users are split according to the files that they may request into G non-intersecting classes with $\frac{K}{G}$ users each. Within the N files, there are N_c common files which may be requested by any user in any class and G subsets of N_u unique files which can only be requested by one class of users. For simplicity, this paper assumes that the number of users and unique files is the same for every class, and that they do not intersect. Therefore, each class has $\frac{K}{G}$ users and N_u unique files, where $N = N_c + GN_u$. Furthermore, we assume that the number of users is smaller than the number of files. The quantities K, G, N_c, and N_u are generally discrete in practice, but this paper will often treat them as continuous to avoid integer effects during calculations. If their values are large enough, the rounding errors can be neglected. This scenario is illustrated in Fig. 1 with only two classes.

In the placement phase, caches are populated with file segments. This paper only considers uncoded prefetching, which means that segments are cached in plain form, not coded together. As asserted in [28], uncoded prefetching is suboptimal, but it has many advantages: it allows for asynchronous transmissions, reduces latency, simplifies the bookeeping, etc. Since file segments are cached in plain form, there will be a section of the cache storing segments from common files and another storing segments from unique files, as shown in Fig. 1.

In the delivery phase, each user k requests a single random file d_k from the server. We denote the probability mass function (pmf) of the random request d_k as $p_{d_k}^{[k]}$.

Definition 1. *The demand set for user k is defined as $S_k \triangleq \left\{n \in \{1, 2, \ldots, N_c + GN_u\} : p_n^{[k]} > 0\right\}$, which represents the set of distinct files that can be requested by user k with a positive probability.*

It can be written that

$$\sum_{\forall d \in S_k} p_d^{[k]} = 1. \tag{1}$$

Definition 2. *The demand vector $\vec{d} = (d_1, \ldots, d_K)$ is defined as the set of files requested by the users in the delivery phase, and $N(\vec{d}) \in [1, min\{K, N\}]$ denotes the number of distinct files in $\vec{d}$.*

Our goal will be to minimize the data rate (traffic from the server to the users) required to satisfy the users' requests. We consider two different metrics for such rate:

Definition 3. *The peak and average rates of a coded caching scheme are respectively defined as*

$$R^*(M) = \max_{\forall \vec{d}: d_k \in S_k} R_{\vec{d}}, \qquad \bar{R} = \sum_{\forall \vec{d}: d_k \in S_k} p_{\vec{d}} R_{\vec{d}}, \tag{2}$$

where $R_{\vec{d}}$ denotes the number of bits transmitted to satisfy request vector $\vec{d}$.

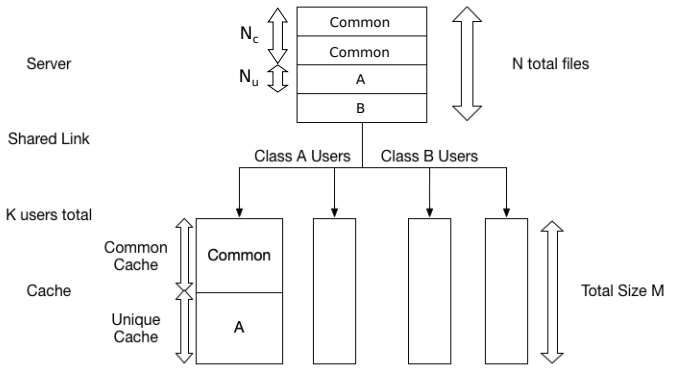

FIGURE 1: System Model with two distinct classes, A and B, each having two users. Each user's cache is divided into a section for common and another for unique files.

The average rate is highly dependent on the pmf of the requests $p_n^{[k]}$. In order to make the equations more tractable and facilitate the comparison with other coded caching schemes, our simulations will focus on the uniform-average rate, defined as follows.

Definition 4. *The uniform-average-rate of a coded caching scheme is defined as*

$$\tilde{R} = \frac{1}{\prod_{k=1}^{K} |S_k|} \sum_{\forall \vec{d}: d_k \in S_k} R_{\vec{d}}, \tag{3}$$

where $|S_k|$ represents the number of distinct files requested by user k with a positive probability.

The uniform-average-rate in Definition 4 is different from that in [30], where the distribution is simply uniform over all the files, namely $[N]^K$. Definition 4 assumes that the distribution is uniform over the demand set for user k. It replaces the joint distribution $p_{\vec{d}}$ with a uniform distribution over the file demand set $S_1 \times S_2 \times \cdots \times S_K$.

B. Maddah-Ali and Niesen's scheme

This paper generalizes the centralized coded caching scheme with uncoded prefetching proposed by Maddah-Ali and Niesen [7], from this point on referred to as MN's scheme, to heterogeneous user profiles. It is therefore important to review such scheme before we go any further.

In the placement phase, MN's scheme splits each file into $\binom{K}{t}$ non-overlapping segments, where $t = \frac{KM}{N}$. Each segment is cached by a distinct set of t users, which results in each user caching $\binom{K-1}{t-1}$ segments per file. Specifically, this scheme is able to satisfy any vector of requests by transmitting at most $\binom{K}{t+1}$ messages of size $\binom{K}{t}^{-1} F$ bits. The peak rate (normalized by the file size F) is written as

$$R_{MN}(K,t) = \frac{\binom{K}{t+1}}{\binom{K}{t}} \tag{4}$$

$$= \frac{K-t}{t+1}. \tag{5}$$

If the server only receives requests for m distinct files (e.g., only some of the users make a request, or their requests overlap), then the transmission rate with MN's scheme will become

$$R(K, \vec{d}, t) = \frac{\binom{K}{t+1} - \binom{K-N(\vec{d})}{t+1}}{\binom{K}{t}}, \tag{6}$$

as was shown in [26].

This paper will treat t as continuous, just like it did with K, G, N_c, and N_u, to avoid integer effects. The next subsection explains how the combinatorial expressions in Eq. (6) can be extended to continuous arguments.

C. Approximation of Transmission Rate for Simulation

This subsection shows how Eq. (6) can be extended into a continuous function over $0 \leq t \leq K$.

When $t \leq 1$, the overall size of all the caches is not enough to store the N files in full. It is therefore necessary to leave a fraction of each file out of the coded caching scheme and transmit it uncoded whenever that file is requested. The minimal such fraction can be found as $p = 1 - \frac{KM}{N}$. Hence, according to [31], the overall transmission rate for demand vector $\vec{d}$ when $t \leq 1$ is

$$R = N(\vec{d})p + [\text{Rate if } t = 1](1-p), \tag{7}$$

where $N(\vec{d})$ denotes the number of distinct files being requested.

When $K-1 < t \leq K$, the opposite happens. The caches are large enough that a fraction of each file needs to be cached by every user, otherwise part of the caches would be left empty. It is therefore never necessary to transmit that fraction and coded caching schemes can be used to transmit the rest when the file is requested. The minimal such fraction can be found as $\gamma = K - t$. Hence, the overall transmission rate when $t \in (K-1, K]$ is

$$R = 0 \cdot \gamma + [\text{Rate if } t = K-1](1-\gamma). \tag{8}$$

When $1 \leq t \leq K-1$ but it is not an integer, Eq. (6) is not well defined because the binomial coefficients require integer and strictly non-negative arguments. In order to interpolate these coefficients continuously, we use the Gamma function, which is defined for all positive real numbers and satisfies $\Gamma(n) = (n-1)!$ for every integer n. Therefore

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} = \frac{\Gamma(n+1)}{\Gamma(k+1)\Gamma(n-k+1)} \tag{9}$$

without error when n and k are integers.

III. Proposed Schemes

The schemes proposed and analyzed in this paper are variations of MN's scheme and were first presented in our conference paper [25].

A. Scheme 1: All common

The system behaves as if all files are common during the placement phase, sacrificing local caching gain in favor of global caching gain. It ignores the distinction between all user profiles and requires every user to cache segments from every file, even if it would never request some of them. MN's scheme with $N = N_c + GN_u$ files is utilized for the placement and delivery. When $N \leq KM \leq (K-1)N$ the peak and average rates can be derived from Eqs. (5) and (6), otherwise it becomes necessary to adjust their values as shown in Eqs. (7) and (8).

- Peak rate: The peak rate is equivalent to that in MN's scheme with N files and K users. According to Eq. (5) it can be computed as:

$$R^{(1)}_{\text{peak}} = \frac{K - t_1}{t_1 + 1}, \tag{10}$$

  where $t_1 = \frac{KM}{N}$.
- Average rate: The average rate is also equivalent to that in MN's scheme with N files and K users. Taking the expectation over the distribution of requests and using Eq. (6) to compute the rate associated with each individual request vector yields:

$$R^{(1)}_{\text{avg}} = \sum_{\forall \vec{d}} p_{\vec{d}} \frac{\binom{K}{t_1+1} - \binom{K-N_1(\vec{d})}{t_1+1}}{\binom{K}{t_1}}, \tag{11}$$

  where $p_{\vec{d}}$ denotes the probability associated to demand vector $\vec{d}$ and $N_1(\vec{d})$ denotes the number of distinct files requested by the K users according to $\vec{d}$.

B. Scheme 2: Split

The system deals with common and unique files separately, decoupling their placement and delivery. A fraction x of each user's cache is devoted to storing segments from common files and the remaining $(1-x)$ to store segments from unique files. Segments from common files are distributed over all K users according to MN's scheme with N_c files and MFx bits of cache per user. Segments from unique files are only cached by the K/G users in their corresponding class, also following MN's scheme to fill the remaining $MF(1-x)$ bits of cache capacity per user. The delivery phase is independent for common and unique files, never encoding segments from different file types in the same message. A clear advantage of this scheme over Scheme 1 is that it reduces the subpacketization (number of segments into which files need to be divided), simplifying the implementation.

If α out of the K/G users in each class request a distinct file from the set of N_u files unique to its class, the peak data rate for this scheme is given by

$$\begin{aligned} R^{(2)}(x,\alpha) &= R_c(x,\alpha) + GR_u(x,\alpha) \\ &= \frac{\binom{K}{t_c+1} - \binom{G\alpha}{t_c+1}}{\binom{K}{t_c}} + G\frac{\binom{\frac{K}{G}}{t_u+1} - \binom{\frac{K}{G}-\alpha}{t_u+1}}{\binom{\frac{K}{G}}{t_u}}, \end{aligned} \tag{12}$$

where $t_c = K\frac{Mx}{N_c}$ and $t_u = \frac{K}{G}\frac{M(1-x)}{N_u}$. The above expressions implicitly assume that t_c and t_u are both larger than 1 and smaller than $K-1$ and $\frac{K}{G}-1$, respectively. Otherwise, they need to be adjusted according to Eqs. (7) or (8). The two terms in Eq. (12) correspond to the rate required to deliver common files, $R_c(x,\alpha)$, and that required to deliver unique files, $R_u(x,\alpha)$, for each class. Despite users are only caching the files in their demand set, x might favor unique files over common files (or vice versa), so the local caching gain is still being sacrificed for the benefit of global caching gain.

- Peak rate: Theorem 7 will later prove that, when the number of users is sufficiently large, the peak rate is achieved when the number of users requesting unique files is the same for every class. Hence, the peak rate can be found as

$$R^{(2)}_{\text{peak}} = \min_x \max_\alpha R^{(2)}(x,\alpha), \tag{13}$$

  where the fraction x is being optimized to minimize the peak rate.
- Average rate: The average rate can be calculated as:

$$R^{(2)}_{\text{avg}} = R_c + \sum_{i=1}^{G} R_{u_i}, \tag{14}$$

  consisting of the average transmission rate for common files R_c and that for unique files R_u in each class. Taking the expectation over the distribution of requests and using Eq. (6) with a reduced memory Mx and number of files N_c to compute the rate associated with each individual request vector yields:

$$R_{\text{c}} = \sum_{\forall \vec{d}} p_{\vec{d}} \frac{\binom{K}{t_c+1} - \binom{K-N_c(\vec{d})}{t_c+1}}{\binom{K}{t_c}}, \tag{15}$$

  where $t_c = \frac{KMx}{N_c}$ and $N_c(\vec{d})$ denotes the number of distinct common files requested by the K users. Similarly, the average transmission rate for unique files in the i-th class can be found by using Eq. (6) with K/G users, N_u files, and capacity for $M(1-x)$ files in the cache:

$$R_{u_i} = \sum_{\forall \vec{d}} p_{\vec{d}} \frac{\binom{\frac{K}{G}}{t_u+1} - \binom{\frac{K}{G}-N_{u_i}(\vec{d})}{t_u+1}}{\binom{\frac{K}{G}}{t_u}}, \tag{16}$$

  where $t_u = \frac{KM(1-x)}{GN_u}$ and $N_{u_i}(\vec{d})$ denotes the number of distinct unique files requested by the K/G users in the i-th class.

C. Scheme 3: All unique

The system behaves as if all files are unique, maximizing local caching gain in detriment of global caching gain. It disregards the fact that common files can be requested by all user classes and independently applies MN's scheme for placement and delivery phases within each class of users. Instead of caching all N_c+GN_u files, the users only cache the N_c+N_u files corresponding to their class during the placement phase. When $G(N_c+N_u) \leq KM \leq (K-G)(N_c+N_u)$ the peak and average rates can be derived from Eqs. (5) and (6), otherwise it becomes necessary to adjust their values as shown in Eqs. (7) and (8).

- Peak rate: The peak rate for each class is equivalent to that in MN's scheme with $\frac{K}{G}$ users and $N_c + N_u$ files. Multiplying the rate in Eqs. (5) by the number of classes G gives:

$$R^{(3)}_{\text{peak}} = G\frac{\frac{K}{G} - t_3}{t_3 + 1}, \tag{17}$$

where $t_3 = \frac{K}{G}\frac{M}{N_c+N_u}$.

- Average rate: The average rate for this scheme can be computed as the sum of the expected rates within each class. Using Eq. (6) with $\frac{K}{G}$ users and $N_c + N_u$ files to compute the expected rates results in:

$$R^{(3)}_{\text{avg}} = \sum_{i=1}^{G}\sum_{\forall \vec{d}} p_{\vec{d}}\frac{\binom{\frac{K}{G}}{t_3+1} - \binom{\frac{K}{G}-N_{3_i}(\vec{d})}{t_3+1}}{\binom{\frac{K}{G}}{t_3}}, \tag{18}$$

where $t_3 = \frac{K}{G}\frac{M}{N_c+N_u}$ and $N_{3_i}(\vec{d})$ represents the number of distinct files requested by the K/G users in the i-th class.

IV. Lower Bound of Peak Rate

This section derives a lower bound for the peak rate of a coded caching system with heterogeneous user profiles and compares it with the peak rate of the three schemes from Section III.

Theorem 1. *The peak rate of a coded caching scheme with G user classes, $\frac{K}{G}$ users in each class, $N = N_c + GN_u$ total files, and cache size of M files per user, can be bounded as*

$$R^*(M) \geq \max_{t\in\{1,\dots,G\}}\max_{s\in\Omega(t)}\left(s\cdot t - \frac{s\cdot t\cdot M}{\left\lfloor\frac{1}{s}\nu(t)\right\rfloor}\right) \tag{19}$$

with $\nu(t) = \left\lfloor\frac{N_c}{t} + N_u\right\rfloor$ and $\Omega(t) = \left\{1,\dots,\min\left(\frac{K}{G},\nu(t)\right)\right\}$. This result is based on a cut-set bound argument.

Proof. Let $t \in \{1,\dots,G\}$, $s \in \Omega(t)$ and consider the first s users from each class $\gamma = 1,2,\dots,t$ denoting their caches $Z_1^\gamma, Z_2^\gamma, \dots, Z_s^\gamma$. Divide the N_c common files into t sets so that each class has $\nu(t)$ files associated with it. Denote them $\{W_1^\gamma, W_2^\gamma, \dots, W_\nu^\gamma\}$, where the argument on ν has been dropped for simplicity. Without loss of generality, we assume that the first s files are requested for every class and the server fulfills those requests by transmitting X_1. The first s users in each class must be able to recover $W_1^\gamma, W_2^\gamma, \dots, W_s^\gamma$ from their caches $Z_1^\gamma, Z_2^\gamma, \dots, Z_s^\gamma$ and X_1. Similarly, when the server sends X_2, the users in each class γ are able to determine $W_{s+1}^\gamma, W_{s+2}^\gamma, \dots, W_{2s}^\gamma$ with their cache $Z_1^\gamma, Z_2^\gamma, \dots, Z_s^\gamma, \gamma = 1,2,\dots,t$. Continue in the same manner up to $X_{\lfloor\frac{1}{s}\nu\rfloor}$. We then have that $X_1, X_2, \dots, X_{\lfloor\frac{1}{s}\nu\rfloor}$ and $Z_1^\gamma, Z_2^\gamma, \dots, Z_s^\gamma$ are enough to determine $W_1^\gamma, W_2^\gamma, \dots, W_{s\lfloor\frac{1}{s}\nu\rfloor}^\gamma$, for $\gamma = 1,2,\dots,t$. Fig. 2 illustrates this setting for $t = 2$.

By the cut-set bound in [32], we can obtain that

$$\left\lfloor\frac{1}{s}\nu(t)\right\rfloor R^*(M) + s\cdot t\cdot M \geq s\cdot t\cdot\left\lfloor\frac{1}{s}\nu(t)\right\rfloor. \tag{20}$$

By solving for $R^*(M)$ and optimizing over all possible choices of s and t, it can be written that

$$R^*(M) \geq \max_{t\in\{1,\dots,G\}}\max_{s\in\Omega(t)}\left(s\cdot t - \frac{s\cdot t\cdot M}{\left\lfloor\frac{1}{s}\nu(t)\right\rfloor}\right), \tag{21}$$

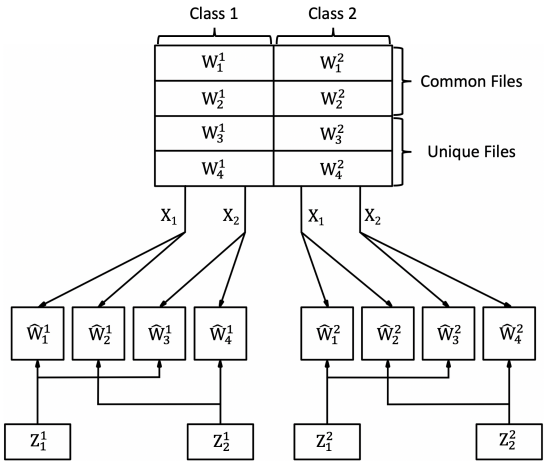

FIGURE 2: Cut corresponding to parameter $s = 2$ in the proof of the converse. In the figure, $N_c = 4, N_u = 2, t = 2, K = 8$.

proving the theorem. □

We use $R_{CB}(s,t)$ to denote the argument maximized in the bound:

$$R_{CB}(s,t) = s\cdot t - \frac{s\cdot t\cdot M}{\left\lfloor\frac{1}{s}\nu(t)\right\rfloor}. \tag{22}$$

Remark. *When $M > N_c + N_u$ the above expression $R_{CB}(s,t)$ is negative for every (s,t), and therefore the bound is trivial. This is to be expected because such M would be sufficient for every user to cache all the files in its demand set.*

Remark. *When $G = 1$, this bound reduces to the one derived in Theorem 2 of [7].*

For simplicity, the rest of this section will assume that the number of files N is larger than the number of users K and that both of them are significantly larger than the number of classes G. This will simplify the proofs by treating $\nu(G)$ as an integer and $\Omega(G)$ as the continuous set $1 \leq s \leq \frac{K}{G}$.

Theorem 2. *For the heterogeneous user profile model with a database of $N = N_c + GN_u$ files, K users with $G << K \leq N$, and a local cache size of M files at each user with $M \leq \frac{N}{2G}$, it can be written that*

$$\frac{R^{(1)}_{\text{peak}}}{R^*(M)} \leq 8. \tag{23}$$

Proof. Loosening the bound in Eq. (19) results in

$$R^*(M) \geq \max_{s\in\Omega(G)} R_{CB}(s,G) \tag{24}$$

$$\geq \max_{s\in\Omega(G)}\left(Gs - \frac{GsM}{\frac{N}{Gs}-1}\right) \tag{25}$$

$$\geq \left(Gs - \frac{GsM}{\frac{N}{Gs}-1}\right), \tag{26}$$

where the last inequality holds for any $1 \leq s \leq \frac{K}{G}$.

We first consider the case when $M \leq \frac{N}{2K}$. Applying the above bound with $s_0 = \frac{K}{2G}$ yields

$$R^*(M) \geq \Big(\frac{K}{2} - \frac{\frac{K}{2}M}{\frac{2N}{K}-1}\Big) \tag{27}$$

$$\geq \frac{K}{2}\Big(\frac{2N-K-KM}{2N-K}\Big) \tag{28}$$

$$\geq \frac{3K}{8}. \tag{29}$$

Clearly, Scheme 1 transmits less than one file per user, so $R^{(1)}_{\text{peak}} \leq K$ and Eq. (23) holds when $M \leq \frac{N}{2K}$.

Next, we consider $\frac{N}{2K} \leq M \leq \max\big(1.5, \frac{N}{K}\big)$. Let $s_1 = \frac{N}{4GM}$ and observe that $1 \leq s_1 \leq \frac{K}{G}$ as long as $K \geq 4G$. Eq. (26) then becomes

$$R^*(M) \geq \Big(Gs_1 - \frac{Gs_1 M}{\frac{N}{Gs_1}-1}\Big) \tag{30}$$

$$\geq N\frac{3M-1}{4M(4M-1)}. \tag{31}$$

When $M \leq \frac{N}{K}$, we cannot apply Eq. (10) directly to find $R^{(1)}_{\text{peak}}$, since $t_1 \leq 1$. Instead, we have to correct it with Eq. (7) obtaining

$$R^{(1)}_{\text{peak}} = \frac{(K-1)KM}{2N} + K\frac{N-KM}{N} \tag{32}$$

$$= \frac{K}{2N}(2N - MK - M). \tag{33}$$

Dividing Eq. (33) by Eq. (31) results in

$$\frac{R^{(1)}_{\text{peak}}}{R^*(M)} \leq \frac{4KM}{N}\cdot\frac{2N-M(K+1)}{2N}\cdot\frac{4M-1}{3M-1} \tag{34}$$

$$\leq 4\cdot 1\cdot 2, \tag{35}$$

where it has been used that $K \leq N$ and therefore $M \geq \frac{1}{2}$.

When $\frac{N}{K} \leq M \leq 1.5$ (which may be an empty set, in which case this part can be ignored), the modification in Eq. (7) is not needed. Dividing Eq. (10) by Eq. (31) and using the fact that $N \geq K$ yields

$$\frac{R^{(1)}_{\text{peak}}}{R^*(M)} \leq \frac{4KM}{KM+N}\cdot\frac{N-M}{N}\cdot\frac{4M-1}{3M-1} \tag{36}$$

$$\leq 4\cdot 1\cdot 2. \tag{37}$$

So far, we have shown that Eq. (23) holds for $M \leq \max\big(1.5, \frac{N}{K}\big)$. It only remains to show that the theorem also holds for $\max(1.5, \frac{N}{K}) \leq M \leq \frac{N}{2G}$. For this, we use $s_2 = \frac{N}{2GM}$ to obtain

$$R^*(M) \geq N\frac{M-1}{2M(2M-1)}. \tag{38}$$

Dividing Eq. (10) by Eq. (38) and imposing $M \geq 1.5$ yields

$$\frac{R^{(1)}_{\text{peak}}}{R^*(M)} \leq \frac{N-M}{N}\cdot\frac{4KM}{KM+N}\cdot\frac{M-\frac{1}{2}}{M-1} \tag{39}$$

$$\leq 1\cdot 4\cdot 2, \tag{40}$$

which proves that $R^{(1)}_{\text{peak}}$ is within a factor of 8 from the optimal for $M \leq \frac{N}{2G}$.

□

Theorem 3. *For the heterogeneous user profile model with K users, a database of $N = N_c + GN_u$ files, and a local cache size of M files at each user with $M \leq \frac{N}{2G}$, it can be written that*

$$\frac{R^{(3)}_{\text{peak}}}{R^*(M)} \leq 8G. \tag{41}$$

Proof. For $M \leq \frac{N}{2K}$, we can use the same argument as in the proof of Theorem 2: $R^* \geq \frac{3K}{8}$ and $R^{(3)}_{\text{peak}} \leq K$, so the theorem holds.

Eq. (31) and Eq. (38) in the proof of Theorem 2 showed that

$$R^*(M) \geq N\frac{3M-1}{4M(4M-1)} \tag{42}$$

for $\frac{N}{2K} \leq M \leq 1.5$ and

$$R^*(M) \geq N\frac{M-1}{2M(2M-1)} \tag{43}$$

for $1.5 \leq M \leq \frac{N}{2G}$. As for $R^{(3)}_{\text{peak}}$, we can use Eq. (17) to compute it when when $M \geq \frac{G}{K}(N_c+N_u)$. Otherwise, the equation needs to be corrected with Eq. (7) resulting in

$$R^{(3)}_{\text{peak}} = K\left(1 - \frac{M(K+G)}{2G(N_c+N_u)}\right) \tag{44}$$

$$\leq K\left(1 - \frac{M(K+G)}{2GN}\right). \tag{45}$$

Hence, when $\frac{N}{2K} \leq M \leq \min\big(1.5, \frac{G}{K}(N_c+N_u)\big)$ we can just divide Eq. (45) by Eq. (42) to obtain

$$\frac{R^{(3)}_{\text{peak}}}{R^*(M)} \leq \left(1 - \frac{M(K+G)}{2GN}\right)\cdot\frac{4KM}{N}\cdot\frac{4M-1}{3M-1} \tag{46}$$

$$\leq \left(1 - \frac{K+G}{4KG}\right)\cdot\frac{4G(N_c+N_u)}{N}\cdot\frac{4M-1}{3M-1} \tag{47}$$

$$\leq 1\cdot 4G\cdot 2. \tag{48}$$

When $1.5 \leq M \leq \frac{G}{K}(N_c+N_u)$ we need to divide by Eq. (43) instead of by Eq. (42), obtaining

$$\frac{R^{(3)}_{\text{peak}}}{R^*(M)} \leq \left(1 - \frac{K+G}{4KG}\right)\cdot\frac{2G(N_c+N_u)}{N}\cdot\frac{2M-1}{M-1} \tag{49}$$

$$\leq 1\cdot 2G\cdot 4. \tag{50}$$

When $\frac{G}{K}(N_c+N_u) \leq M \leq 1.5$ we need to divide Eq. (17) by Eq. (42), obtaining

$$\frac{R^{(3)}_{\text{peak}}}{R^*(M)} \leq \frac{N_c+N_u-M}{N}\cdot\frac{4GKM}{KM+G(N_c+N_u)}\cdot\frac{4M-1}{3M-1} \tag{51}$$

$$\leq 1\cdot 4G\cdot 2. \tag{52}$$

Finally, when $\max\big(1.5, \frac{G}{K}(N_c+N_u)\big) \leq M \leq \frac{N}{2G}$, we divide Eq. (17) by Eq. (43) obtaining

$$\frac{R^{(3)}_{\text{peak}}}{R^*(M)} \leq \frac{N_c+N_u-M}{N}\cdot\frac{2GKM}{KM+G(N_c+N_u)}\cdot\frac{2M-1}{M-1} \tag{53}$$

$$\leq 1\cdot 2G\cdot 4. \tag{54}$$

□

Theorem 4. *For the heterogeneous user profile model with K users, a database of $N = N_c + GN_u$ files, and a local cache size of M files at each user with $\frac{G}{K}(N_c + N_u) \leq M \leq \frac{N}{2G}$, it can be written that*

$$\frac{R^{(2)}_{\text{peak}}}{R^*(M)} < 8(G+1). \tag{55}$$

Proof. The peak rate of Scheme 2 can be bound as follows.

$$R^{(2)}_{\text{peak}} = \min_x \max_\alpha R^{(2)}(x,\alpha) \tag{56}$$

$$\leq \max_\alpha R^{(2)}\left(x = \frac{N_c}{N}, \alpha\right) \tag{57}$$

$$\leq \max_\alpha R_c\left(x = \frac{N_c}{N}, \alpha\right) + \max_\alpha GR_u\left(x = \frac{N_c}{N}, \alpha\right) \tag{58}$$

$$\leq R_c\left(x = \frac{N_c}{N}, \alpha = 0\right) + GR_u\left(x = \frac{N_c}{N}, \alpha = \frac{K}{G}\right) \tag{59}$$

$$\leq R^{(1)}_{\text{peak}} + G\frac{\frac{K}{G} - t_1}{t_1 + 1}, \tag{60}$$

where $N = N_c + GN_u$ and $t_1 = \frac{KM}{N}$. Since $t_3 \leq t_1$ and $R^{(3)}_{\text{peak}}$ decreases monotonically with t_3 we can conclude that

$$R^{(2)}_{\text{peak}} \leq R^{(1)}_{\text{peak}} + R^{(3)}_{\text{peak}}. \tag{61}$$

Finally, we apply Theorems 2 and 3 to obtain Eq. (55). □

We do not attempt to characterize a bound for average rate because it would depend on the popularity distribution of the files. A bound for uniform-average rate could be derived, but we believe that it would not provide valuable insights for the general case.

V. Results

This first part of this section studies how to optimize the distribution of cache between common and unique files in Scheme 2 so that the peak rate and uniform-average rate are minimized. We discover that when users' cache storage is small, devoting all the cache to common files will minimize both the peak and the average rate of transmission. The second part of the section provides detailed comparisons between the peak rates of the three schemes proposed in Section III and analyzes which scheme offers the best performance for each value of M. A partial summary of results can be found in Table I.

Our previous conference paper [1] provided some partial and asymptotic results for uniform-average rate, but we have decided not to include those here, postponing them to future work on a separate paper.

A. Optimizing x for Scheme 2

When M is large, users are able to cache most of the files and the choice of x is less relevant. Furthermore, scenarios where caches are almost as large as the whole library rarely come up in practical applications. Hence, we will focus our analysis on the case with relatively small M compared with the size of the library N.

Theorem 5. *There exists a $\theta \gg 0$ such that for all $K > \theta$ and $M \leq \frac{1}{K}\min(N_c, GN_u)$, the uniform-average rate of Scheme 2 is minimized by devoting all the cache to either common or unique files. Specifically, it should all be devoted to common files ($x = 1$) when*

$$\frac{N_u}{N_c} > G \cdot \frac{\mathbb{E}^2[N_u(\vec{d})] + \mathbb{E}[N_u(\vec{d})]}{\mathbb{E}^2[N_c(\vec{d})] + \mathbb{E}[N_c(\vec{d})]}, \tag{62}$$

otherwise it should all be devoted to unique files ($x = 0$). In Eq. (62),

$$\mathbb{E}[N_c(\vec{d})] = N_c\left[1 - \left(\frac{N_c + N_u - 1}{N_c + N_u}\right)^K\right], \tag{63}$$

$$\mathbb{E}[N_u(\vec{d})] = N_u\left[1 - \left(\frac{N_c + N_u - 1}{N_c + N_u}\right)^{K/G}\right]. \tag{64}$$

Proof. See Appendix. □

Theorem 5 generalizes Prop. 3 from paper [28], where there exist two classes with one user each. When the number of common files is large and the cache size is below half of the common files, the users should only cache common files.

Corollary 1. *If the users' devices have relatively small storage and the number of common files is not too large, it is recommended for the users to devote all their cache to common files and transmit the unique files uncoded.*

B. Peak Rate Comparison

First, we compare the peak rate of Schemes 1 and 3, since they have the simplest expressions.

Theorem 6. *When M is small, Scheme 1 offers lower peak rate than Scheme 3, and vice versa. Specifically,*

$$R^{(1)}_{\text{peak}} \leq R^{(3)}_{\text{peak}} \qquad \Leftrightarrow \qquad M \leq N_c - \frac{GN_u}{K}. \tag{65}$$

Proof. This theorem can be proved by simple manipulation of Eqs (10) and (17). □

Corollary 2. *When M is small, it is often beneficial for users to cache segments from undesired files (unselfish caching), to increase multicasting opportunities. The loss in local caching gain is more than compensated by the gain in global caching gain [33]. In fact, it was recently proved that MN's scheme can provide unbounded gains over selfish coded caching [34].*

In Schemes 1 and 3, the number of users requesting common versus unique files is irrelevant, since segments from both files can be encoded together. In Scheme 2, however, it plays a major role. We now intend to show that in order to compute the peak rate, we only need to consider the case where the subdivision is the same for all user classes.

Theorem 7. *There exists a number $\alpha \in (0, \frac{K}{G})$ such that the peak rate for Scheme 2 is achieved when every class has α (or, occasionally, $\alpha+1$ due to discretization constraints) users requesting unique files.*

Proof. Let α_i represent the number of users from class i requesting unique files, and assume that $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_G)$ maximizes the rate, given by

$$R(\boldsymbol{\alpha}) = R_c\left(\frac{1}{G}\sum_{i=1}^{G}\alpha_i\right) + \sum_{i=1}^{G} R_u(\alpha_i), \tag{66}$$

where R_c and R_u have been defined by Eq. (12) and we omit x for simplicity.

Without loss of generality, assume $\alpha_1 > \alpha_2$ and let $\boldsymbol{\beta} = (\alpha_1 - 1, \alpha_2 + 1, \alpha_3, \ldots, \alpha_G)$. Then

$$\begin{aligned} R(\boldsymbol{\beta}) - R(\boldsymbol{\alpha}) = \\ R_u(\alpha_1 - 1) - R_u(\alpha_1) + R_u(\alpha_2 + 1) - R_u(\alpha_2). \end{aligned} \tag{67}$$

We now prove that $R(\boldsymbol{\beta}) - R(\boldsymbol{\alpha}) \geq 0$ or, equivalently,

$$R_u(\alpha_2 + 1) - R_u(\alpha_2) \geq R_u(\alpha_1) - R_u(\alpha_1 - 1). \tag{68}$$

This result follows from the fact that the rate is submodular in the number of requests, but we prove it anyway. With $\alpha_1 > \alpha_2$, Eq. (68) can be written as

$$\frac{\binom{\frac{K}{G}-\alpha_2}{t_u+1} - \binom{\frac{K}{G}-\alpha_2-1}{t_u+1}}{\binom{\frac{K}{G}}{t_u}} \geq \frac{\binom{\frac{K}{G}-\alpha_1+1}{t_u+1} - \binom{\frac{K}{G}-\alpha_1}{t_u+1}}{\binom{\frac{K}{G}}{t_u}} \tag{69}$$

$$\binom{\frac{K}{G} - (\alpha_2 + 1)}{t_u} \geq \binom{\frac{K}{G} - \alpha_1}{t_u}, \tag{70}$$

which is true, since binomial coefficients increase monotonically with the number of elements.

Therefore, for any general $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_G)$ where $\alpha_i > \alpha_j$, if we shift one user who requests a unique file from class i to class j, we will obtain a larger or equal new transmission rate than the original rate. By repeating the shifting process as many times as necessary, we can find a maximal transmission rate with all classes having nearly the same number of users requesting unique files. Strictly speaking, they could be off by a single user, but for large enough K (*i.e.* using the continuous relaxation of the problem), we can conclude that a uniform set of coefficients would achieve the peak rate. □

In the proof of Theorem 7, we utilized the method of *Reductio ad absurdum*, namely showing that the opposite scenario would lead to contradiction, to prove that Scheme 2 can achieve the peak rate of transmission when each class has an equal number of users who request unique files. We are now ready to compare the peak rate of Scheme 2 with that for the other two.

Theorem 8. *When M is large enough, Scheme 2 offers lower peak rate than Scheme 3. Specifically,*

$$R_{\text{peak}}^{(2)} \leq R_{\text{peak}}^{(3)} \qquad \Leftarrow \qquad M \geq \frac{G}{G-1}\frac{K+1}{K}N_u. \tag{71}$$

Proof. If $x = 1 - \frac{N_u}{M}$, each user stores all the unique files that it might request. The worst case α is therefore $\alpha = 0$. Observe that

$$\begin{aligned} R_{\text{peak}}^{(2)} &= \min_x \max_\alpha R^{(2)}(x, \alpha) && (72)\\ &\leq \max_\alpha R^{(2)}\left(1 - \frac{N_u}{M}, \alpha\right) && (73)\\ &= R^{(2)}(1 - \frac{N_u}{M}, 0) && (74)\\ &= K\frac{N_u + N_c - M}{K(M - N_u) + N_c}. && (75) \end{aligned}$$

After some rearrangement, Eq. (17) can be written as

$$R_{\text{peak}}^{(3)} = KG\frac{N_c + N_u - M}{KM + G(N_c + N_u)}. \tag{76}$$

A simple comparison of the last two equations yields Eq. (71). □

Theorem 9. *When M is large enough, Scheme 2 provides lower peak rate than Scheme 1. Specifically,*

$$R_{\text{peak}}^{(2)} \leq R_{\text{peak}}^{(1)} \Leftarrow M \geq \max\left(\frac{G}{G-1}\frac{K+1}{K}N_u, \frac{N_c}{G} + N_u\right). \tag{77}$$

Proof. From Eq. (75) we can observe that

$$R_{\text{peak}}^{(2)} \leq K\frac{N_u + N_c - M}{K(M - N_u) + N_c}, \tag{78}$$

and Eq. (10) can be rearranged as

$$R_{\text{peak}}^{(1)} = \frac{K - \frac{KM}{N_c + GN_u}}{\frac{KM}{N_c + GN_u} + 1}. \tag{79}$$

By comparing these two equations, we are able to generate Eq. (77). □

Corollary 3. *Scheme 2 provides the lowest peak rate of the three when $M \geq \max\left(\frac{G}{G-1}\frac{K+1}{K}N_u, \frac{N_c}{G} + N_u\right)$.*

Proof. This corollary can be simply proven by combining Theorem 8 and Theorem 9, setting M to be the larger value between the two. □

Theorem 10. *When $K \geq (G-1)\left(1 + \frac{GN_u}{N_c}\right)$ and M is small enough, Scheme 1 offers lower peak rate than Scheme 2. Specifically,*

$$R_{\text{peak}}^{(1)} \leq R_{\text{peak}}^{(2)} \qquad \Leftarrow \qquad M \leq \frac{\min(N_c, GN_u)}{K}. \tag{80}$$

Proof. If $M \leq \frac{\min(N_c, GN_u)}{K}$ then $t_1 \leq 1$ and we can combine Eqs. (10) and (7) to obtain

$$R_{\text{peak}}^{(1)} = K - \frac{M}{2}\frac{K(K+1)}{N_c + GN_u}, \tag{81}$$

As for $R_{\text{peak}}^{(2)}$, it is defined as the highest rate experienced for any number of unique requests α:

$$\begin{aligned} R_{\text{peak}}^{(2)} &= \min_x \max_\alpha R^{(2)}(x, \alpha) && (82)\\ &\geq \min_{0 \leq x \leq 1} R^{(2)}(x, \alpha_0) && (83) \end{aligned}$$

M range	Result
$M \leq \frac{\min(N_c, GN_u)}{K}$, $K \geq \frac{(G-1)N}{N_c}$	$R^{(1)}_{\text{peak}}$ best
$M \geq \max\left(\frac{G}{G-1}\frac{K+1}{K}N_u, \frac{N}{G}\right)$	$R^{(2)}_{\text{peak}}$ best
$M \leq \frac{N}{2G}$	$R^{(1)}_{\text{peak}}/R^*(M) \leq 8$
$M \leq \frac{N}{2G}$	$R^{(2)}_{\text{peak}}/R^*(M) < 8(G+1)$
$M \leq \frac{N}{2G}$	$R^{(3)}_{\text{peak}}/R^*(M) \leq 8G$

TABLE I: Summary of peak rate results

for any specific value of α_0. Combining Eqs. (12) and (7) yields

$$R^{(2)}(x,\alpha) = \frac{\binom{K}{2} - \binom{G\alpha}{2}}{\binom{K}{1}} t_c + (K - G\alpha)(1 - t_c) \quad (84)$$
$$+ G\frac{\binom{\frac{K}{G}}{2} - \binom{\frac{K}{G}-\alpha}{2}}{\binom{\frac{K}{G}}{1}} t_u + G\alpha(1-t_u),$$

where $t_c = \frac{KMx}{N_c}$ and $t_u = \frac{KM(1-x)}{GN_u}$ are both smaller than 1. Since $R^{(2)}(x,\alpha)$ is a linear function of x, it will achieve its minimum at either $x = 0$ or $x = 1$. Consequently,

$$R^{(2)}_{\text{peak}} \geq \min\left[R^{(2)}(x=0,\alpha), R^{(2)}(x=1,\alpha)\right]$$
$$\geq K - \frac{M}{2}\max\left[G\frac{\alpha^2+\alpha}{N_u}, \frac{(G\alpha-K)^2 - G\alpha + K}{N_c}\right]$$

for any value of α. It only remains to find a value for α such that

$$G\frac{\alpha^2+\alpha}{N_u} \leq \frac{K(K+1)}{N_c + GN_u} \quad (85)$$
$$\frac{(G\alpha-K)^2 - G\alpha + K}{N_c} \leq \frac{K(K+1)}{N_c + GN_u}. \quad (86)$$

We propose using $\alpha_0 = \frac{K}{G}\frac{GN_u}{N_c+GN_u}$, which when plugged into Eqs. (85) and (86) results in

$$\frac{K}{N_c+GN_u}\left(G - 1 - K\frac{N_c}{Nc+GN_u}\right) \leq 0 \quad (87)$$
$$-\frac{K^2GN_u}{(N_c+GN_u)^2} \leq 0, \quad (88)$$

respectively. Since $K \geq (G-1)\left(1 + \frac{GN_u}{N_c}\right)$, both inequalities hold and the theorem is proved. □

Corollary 4. *For small enough M, Scheme 1 yields lower peak rate than Schemes 2 and 3. For large enough M, Scheme 2 offers the lowest rate of the three. In some cases, there is a range of intermediate M values for which Scheme 3 has lower rate than the other two.*

VI. Numerical Simulations

This section provides simulations illustrating the peak and uniform-average rates of the three proposed schemes, as well as lower bounds to provide a framework for comparison. To the extent of our knowledge, there are no schemes in the literature which could be suitable for our scenario with heterogeneous user profiles. The best performing schemes for homogeneous users are those found by solving combinatorial optimization problems as described in in [13] and [35]. However, with uniform file popularities, cache capacities, and file sizes, those schemes are equivalent to Maddah-Ali and Niesen's scheme. Solving the optimization problems while ignoring user classes results in Scheme 1 and doing it independently for each class results in Scheme 3.

Fig. 3 illustrates the peak rate of the three schemes and the cut set bound, for different cache sizes M and number of classes G. The number of users per class is set as $K/G = 8$ in all cases. The results match the statement in Corollary 4: it is better to use Scheme 1 (all common) for small cache sizes and Scheme 2 (split) for large ones, regardless of the number of classes. This result seems counter-intuitive, since it suggests that every user should cache segments from every file when the caches are small, even as the number of unique files scales with the number of classes. However, it turns out that the multicasting gain more than compensates for the loss in local caching.

The peak rate values increase with the number of classes due mainly to the increase in the number of files and users. It can be seen that the peak rate of Scheme 3 (all unique) increases above the others, reaching a point when it is never the preferred option. Again, this is somewhat counter-intuitive; it seems like a good idea to deal with each class independently when the number of classes is large, but it is not.

Fig. 4 shows the uniform-average rates of the three schemes in the same scenario. The minimal uniform-average rate in scenarios with heterogeneous user profiles is unknown, so we define a new scheme "MN with oracle" to provide an approximate lower bound. In this scheme the system knows in advance which users will request common and unique files, and it populates their caches using MN's scheme for common and unique files separately. This results in the following uniform-average rate:

$$R_{\text{orc}} = \sum_{k_c=0}^{K} p_{k_c}\Bigg(E\left[R(k_c, m, t_{oc})\right]$$
$$+ G \cdot E\left[R\left(\frac{K-k_c}{G}, m, t_{ou}\right)\right]\Bigg),$$

where k_c represents the number of users that request common files, $E[R(K,m,t)]$ is the expectation of the rate defined in Eq. (6) over the number of distinct files requested m, $t_{oc} = \frac{k_c M}{N_c}$, $t_{ou} = \frac{(K-k_c)M}{GN_u}$, and

$$p_{k_c} = \binom{K}{k_c}\left(\frac{N_c}{N_c+N_u}\right)^{k_c}\left(\frac{N_u}{N_c+N_u}\right)^{K-k_c} \quad (89)$$

is the probability that k_c users request common files.

The results suggest that Scheme 2, which splits the placement and delivery of common and unique files, is highly suboptimal when the number of classes is small, unless the cache memory is very small or very large. However, when the number of classes increases, Scheme 2 achieves the lowest average rate among all three of the schemes. This result aligns with Prop. 3 in [1].

It is worth noting that these results are different from those previously observed for the peak rate: for small M, Scheme 2

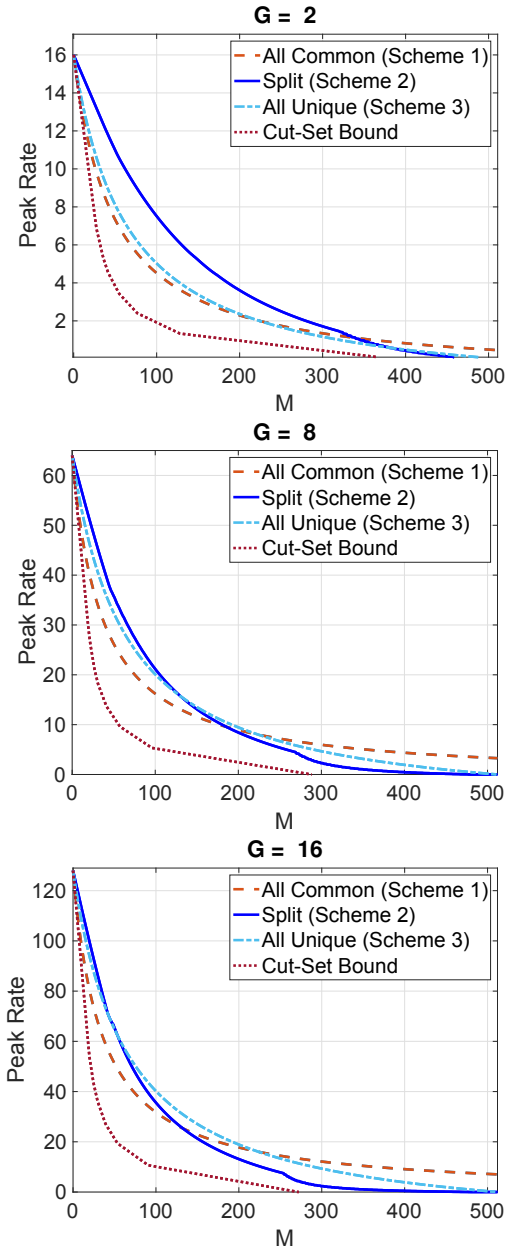

FIGURE 3: Peak rates and cut-set bound vs cache size (M) for $N_c = 256$, $N_u = 256$, and 8 users per class.

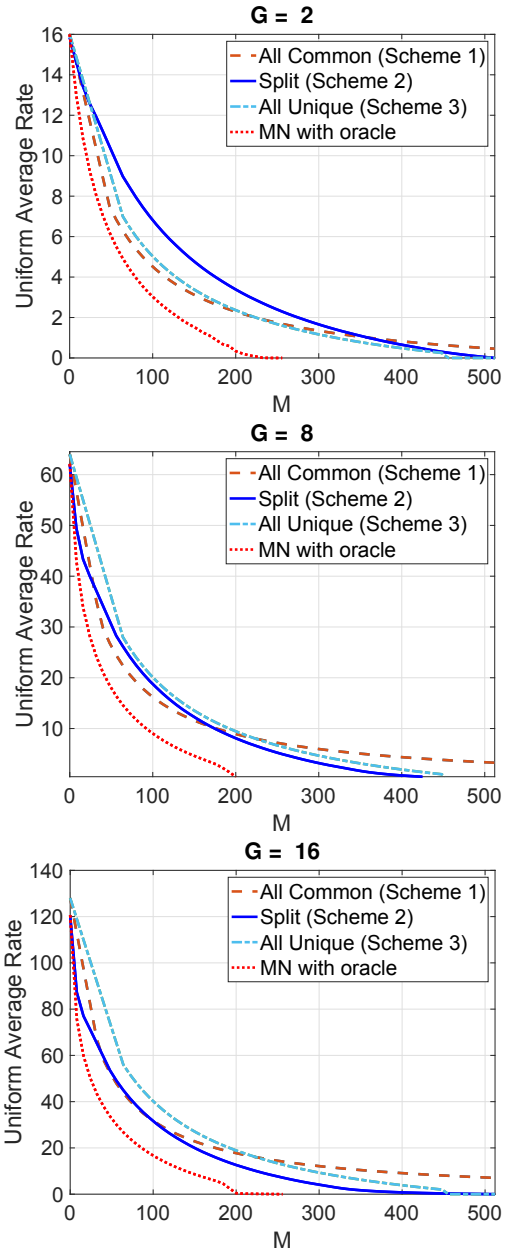

FIGURE 4: Average rate vs Cache size (M) for $N_c = 256$, $N_u = 256$, and 8 users per class.

presents the lowest uniform-average rate and the highest peak rate among the three schemes, regardless of the number of classes. Even though Scheme 2 does not provide the lowest peak rate of transmission when the users' storage is small, we should still consider Scheme 2 as a primary choice in this scenario. As in practical systems it is common for different users to request the same files, the average rate of transmission can represent the traffic of a coded caching system better than the peak rate.

Fig. 5 investigates the performance of the three schemes as the number of classes grows. In this scenario, $N_c = N_u = 256$, there are 8 users in each class and we set $M = 256$ to provide enough storage for each user to cache half of the files it could request. The top plot stands for the comparison of the peak rates of three schemes, and the bottom plot compares the average rates of three schemes. As the number of classes increases, so does the total number of files and users. This results in worsening performance for all three schemes. Schemes 1 (all common) and 3 (all unique) suffer nearly linear degradation, while Scheme 2 (split) scales better. This is because Scheme 2 (split) is able to adjust its cache distribution as the number of classes increases so that both peak and average rate are minimized. Therefore, when there are more classes of users joining the network, Scheme 2

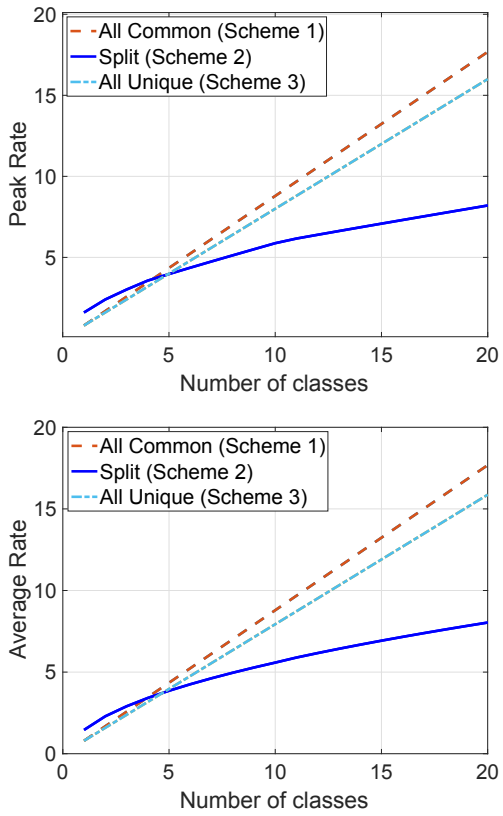

FIGURE 5: Evolution of the peak rate (top) and the average rate (bottom) as the number of classes G grows in a system with $N_c = 256$, $N_u = 256$, $M = 256$, and 8 users per class.

(split) is able to provide better service than the other schemes. Fig. 5 also illustrates the benefit of user profiling. A service provider ignoring user preferences would experience the peak and average rates shown for Scheme 1 (all common), while those taking the time to classify the users into G profile classes could reduce those rates to the ones shown for Schemes 2 and 3. When the number of classes (and therefore files and users) is large, the difference can be significant. Similar results for $N_c = 64$, $N_u = 64$ and $M = 64$ can be found in papers [25] and [1].

VII. Conclusion

This paper proposes three coded caching schemes with uncoded pre-fetching which are suitable for a system where end users are categorized into classes according to their demand distributions. It is assumed that the files are either common, which means that they can be requested by any user in any class, or unique, meaning that only users in a specific class are likely to request them. The first scheme treats all files as if they were common, the second one decouples the delivery of common and unique files, and the third treats all files as if they were unique.

The peak and uniform-average rates of the three schemes are derived and compared with each other, showing that there exist conditions under which each scheme outperforms the other two. Specifically, Scheme 1 provides the lowest peak rate when the caches are small and Scheme 2 when the caches are large. Scheme 3 is best for intermediate cache sizes when the number of classes is small and the number of users is large. Their peak rates are also compared with a cut-set lower bound on the achievable rate to obtain bounds on the gap to optimality for each scheme. We also proposed cache distribution strategies so that both peak rate and average rate of transmission are minimized for Scheme 2.

Our system model assumed that the file preferences are different for each group of users. However, in practical systems there exists overlap between different groups. Hence, inspired by the choice of cache distribution, we acknowledge that it is possible and valuable to explore the effect that the number of classes, namely G, has on the rate of transmission. In future work, we plan to study this effect as well as the uniform-average rate of the three schemes in more detail.

References

[1] C. Zhang and B. Peleato, "Average rate for coded caching with heterogeneous user profiles," *IEEE International Conf. on Comm*, June 2020.
[2] S. Yi, C. Li, and Q. Li, "A survey of fog computing: concepts, applications and issues," in *Proceedings of the 2015 workshop on mobile big data*, 2015, pp. 37–42.
[3] P. Duan, Y. Jia, L. Liang, J. Rodriguez, K. M. S. Huq, and G. Li, "Space-reserved cooperative caching in 5g heterogeneous networks for industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 6, pp. 2715–2724, 2018.
[4] S. P. Shariatpanahi, G. Caire, and B. H. Khalaj, "Physical-layer schemes for wireless coded caching," *IEEE Transactions on Information Theory*, vol. 65, no. 5, pp. 2792–2807, 2018.
[5] M. Salehi, A. Tolli, S. P. Shariatpanahi, and J. Kaleva, "Subpacketization-rate trade-off in multi-antenna coded caching," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.
[6] R. Sun, H. Zheng, J. Liu, X. Du, and M. Guizani, "Placement delivery array design for the coded caching scheme in medical data sharing," *Neural Computing and Applications*, vol. 32, no. 3, pp. 867–878, 2020.
[7] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Transactions on Information Theory*, vol. 60, no. 5, pp. 2856–2867, 2014.
[8] U. Niesen and M. A. Maddah-Ali, "Coded caching with nonuniform demands," *IEEE Transactions on Information Theory*, vol. 63, no. 2, pp. 1146–1158, 2017.
[9] R. Pedarsani, M. A. Maddah-Ali, and U. Niesen, "Online coded caching," *IEEE/ACM Transactions on Networking (TON)*, vol. 24, no. 2, pp. 836–845, 2016.
[10] C.-H. Chang and C.-C. Wang, "Coded caching with full heterogeneity: Exact capacity of the two-user/two-file case," in *2019 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2019, pp. 6–10.
[11] A. M. Ibrahim, A. A. Zewail, and A. Yener, "Coded caching for heterogeneous systems: An optimization perspective," *IEEE Transactions on Communications*, vol. 67, no. 8, pp. 5321–5335, 2019.
[12] M. Bayat, K. Wan, M. Ji, and G. Caire, "Cache-aided modulation for heterogeneous coded caching over a gaussian broadcast channel," in *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE, 2020, pp. 1–6.
[13] A. M. Daniel and W. Yu, "Optimization of heterogeneous coded caching," *IEEE Transactions on Information Theory*, vol. 66, no. 3, pp. 1893–1919, 2019.
[14] J. Zhang, X. Lin, and X. Wang, "Coded caching under arbitrary popularity distributions," *IEEE Transactions on Information Theory*, vol. 64, no. 1, pp. 349–366, 2018.
[15] N. Golrezaei, A. F. Molisch, A. G. Dimakis, and G. Caire, "Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution," *IEEE Communications Magazine*, vol. 51, no. 4, pp. 142–149, 2013.
[16] N. Karamchandani, U. Niesen, M. A. Maddah-Ali, and S. N. Diggavi, "Hierarchical coded caching," *IEEE Transactions on Information Theory*, vol. 62, no. 6, pp. 3212–3229, 2016.

[17] K. Wan, M. Cheng, M. Kobayashi, and G. Caire, "On the optimal load-memory tradeoff of coded caching for location-based content," *IEEE Transactions on Communications*, vol. 70, pp. 3047–3062, 2022.

[18] J. Zhang, X. Lin, C.-C. Wang, and X. Wang, "Coded caching for files with distinct file sizes," in *2015 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2015, pp. 1686–1690.

[19] A. Hannak, P. Sapiezynski, A. Molavi Kakhki, B. Krishnamurthy, D. Lazer, A. Mislove, and C. Wilson, "Measuring personalization of web search," in *Proceedings of the 22nd international conference on World Wide Web*. ACM, 2013, pp. 527–538.

[20] A. McCallum, K. Nigam, J. Rennie, and K. Seymore, "A machine learning approach to building domain-specific search engines," in *IJCAI*, vol. 99. Citeseer, 1999, pp. 662–667.

[21] E. Agichtein, E. Brill, S. Dumais, and R. Ragno, "Learning user interaction models for predicting web search result preferences," in *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2006, pp. 3–10.

[22] Y. Lu, W. Chen, and H. V. Poor, "On the effective throughput of coded caching: A game theoretic perspective," *IEEE Transactions on Communications*, vol. 69, pp. 1387–1402, 2021.

[23] J. He, C. Li, and L. Song, "Coded caching with heterogeneous user groups," in *2020 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*. IEEE, 2020, pp. 1–6.

[24] B. Tegin and T. M. Duman, "Coded caching with user grouping over wireless channels," *IEEE Wireless Communications Letters*, 2020.

[25] S. Wang and B. Peleato, "Coded caching with heterogeneous user profiles," *IEEE Internat. Symp. on Information Theory (ISIT)*, 2019.

[26] T. Luo, V. Aggarwal, and B. Peleato, "Coded caching with distributed storage," *IEEE Transactions on Information Theory*, vol. 65, no. 12, pp. 7742–7755, 2019.

[27] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "The exact rate-memory tradeoff for caching with uncoded prefetching," *IEEE Transactions on Information Theory*, vol. 64, no. 2, pp. 1281–1296, 2018.

[28] C.-H. Chang, C.-C. Wang, and B. Peleato, "On coded caching for two users with overlapping demand sets," *IEEE International Conf. on Comm*, June 2020.

[29] C.-H. Chang, B. Peleato, and C.-C. Wang, "Coded caching with full heterogeneity: Exact capacity of the two-user/two-file case," *IEEE Transactions on Information Theory*, pp. 1–1, 2022.

[30] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Characterizing the rate-memory tradeoff in cache networks within a factor of 2," *IEEE Transactions on Information Theory*, vol. 65, no. 1, pp. 647–663, 2018.

[31] T. Luo and B. Peleato, "The transfer load-i/o trade-off for coded caching," *IEEE Communications Letters*, vol. 22, no. 8, pp. 1524–1527, 2018.

[32] C. TM and T. JA, "Elements of information theory," *Wiley Series in Telecommunications*, 1991.

[33] C.-H. Chang and C.-C. Wang, "Coded caching with heterogeneous file demand sets - the insufficiency of selfish coded caching," *IEEE Internat. Symp. on Information Theory (ISIT)*, 2019.

[34] F. Brunero and P. Elia, "Unselfish coded caching can yield unbounded gains over symmetrically selfish caching," *arXiv preprint arXiv:2109.04807*, 2021.

[35] S. Jin, Y. Cui, H. Liu, and G. Caire, "Structural properties of uncoded placement optimization for coded delivery," *arXiv preprint arXiv:1707.07146*, 2017.

APPENDIX

A. Proof of Theorem 5

Proof. When $M \leq \frac{1}{K}\min(N_c, GN_u)$, we are assured that both t_c and t_u in Eqs. (15) and (16) will be smaller than 1. As a consequence, the uniform-average rate must be adjusted according to Eq. (7) and $R^{(2)}_{\text{avg}}$ becomes a linear function of x (Eq. (7) is a linear function of p, which is in turn a linear function of x). Since it is only defined over $0 \leq x \leq 1$, $R^{(2)}_{\text{avg}}$ must be minimized by either $x = 0$ or $x = 1$, depending on the sign of its partial derivative respect to x.

The law of large numbers tells us that when the number of users K is large, the number of distinct files requested will be very close to its expected value for almost every demand vector $\vec{d}$. If we approximate $N_c(\vec{d})$ and $N_u(\vec{d})$ with their expected values, we have

$$\frac{\partial R^{(2)}_{\text{avg}}}{\partial x} = \frac{\partial R_c}{\partial t_c}\frac{\partial t_c}{\partial x} + G\frac{\partial R_u}{\partial t_u}\frac{\partial t_u}{\partial x} \tag{90}$$

$$= \sum_{\forall \vec{d}} p_{\vec{d}} \left(\frac{KM}{N_c}\left\{ -N_c(\vec{d}) + \frac{\binom{K}{2} - \binom{K-N_c(\vec{d})}{2}}{K} \right\} + \frac{KM}{N_u}\left\{ N_u(\vec{d}) - \frac{\binom{\frac{K}{G}}{2} - \binom{\frac{K}{G}-N_u(\vec{d})}{2}}{\frac{K}{G}} \right\} \right) \tag{91}$$

$$\simeq \frac{KM}{N_c}\left\{ -\mathbb{E}[N_c(\vec{d})] + \frac{\binom{K}{2} - \binom{K-\mathbb{E}[N_c(\vec{d})]}{2}}{K} \right\} + \frac{KM}{N_u}\left\{ \mathbb{E}[N_u(\vec{d})] - \frac{\binom{\frac{K}{G}}{2} - \binom{\frac{K}{G}-\mathbb{E}[N_u(\vec{d})]}{2}}{\frac{K}{G}} \right\}. \tag{92}$$

After expanding Eq. (92) and cancelling out terms we find that $\frac{\partial R^{(2)}_{\text{avg}}}{\partial x}$ is negative (*i.e.* , $x = 1$ minimizes $R^{(2)}_{\text{avg}}$) when

$$\frac{N_u}{N_c} > \frac{\mathbb{E}[N_u(\vec{d})]}{\mathbb{E}[N_c(\vec{d})]} \cdot \frac{\frac{1}{2} - \frac{G}{2K}\left(\frac{K}{G} - \mathbb{E}[N_u(\vec{d})] - 1\right)}{\frac{1}{2} - \frac{1}{2K}\left(K - \mathbb{E}[N_c(\vec{d})] - 1\right)} > G \cdot \frac{\mathbb{E}^2[N_u(\vec{d})] + \mathbb{E}[N_u(\vec{d})]}{\mathbb{E}^2[N_c(\vec{d})] + \mathbb{E}[N_c(\vec{d})]}, \tag{93}$$

and positive otherwise (*i.e.* , $x = 0$ minimizes $R^{(2)}_{\text{avg}}$).

The expected number of distinct common and unique files requested can be computed defining a binary variable for each file (taking value 1 if at least one user requests it and zero otherwise) and using the method of indicators:

$$\mathbb{E}[N_c(\vec{d})] = N_c\left[1 - \left(\frac{N_c + N_u - 1}{N_c + N_u}\right)^K\right], \tag{94}$$

$$\mathbb{E}[N_u(\vec{d})] = N_u\left[1 - \left(\frac{N_c + N_u - 1}{N_c + N_u}\right)^{K/G}\right]. \tag{95}$$

□

Ciyuan Zhang received the B.S. degrees in Electrical Engineering and Automation from Xi'an Jiaotong University, Xi'an, China, in 2016, and the M.S. in Electrical Engineering from Columbia University, New York, USA, in 2018. He currently is a Ph.D. candidate in the Electrical and Computer Engineering at Purdue University, West Lafayette, IN, USA.

His research interests include automatic control, epidemic process, coded caching, and convex optimization.

Su Wang is a Ph.D. student in Electrical and Computer Engineering at Purdue University. He received his B.S. degree in Electrical Engineering from Purdue University in 2018.

Vaneet Aggarwal received the B.Tech. degree in 2005 from the Indian Institute of Technology, Kanpur, India, and the M.A. and Ph.D. degrees in 2007 and 2010, respectively from Princeton University, Princeton, NJ, USA, all in Electrical Engineering. He was a Senior Member of the Technical Staff Research with AT&T Labs Research, Bedminster, NJ, USA, from 2010 to 2014. He was an Adjunct Assistant Professor at Columbia University, NY from 2013-2014, and an Adjunct Professor at IISc Bangalore, India from 2018-2019, and is currently a Visiting Professor at KAUST, Saudi Arabia. He is currently a Full Professor at Purdue University, West Lafayette, IN, USA, where he has been since Jan 2015. His current research interests are in machine learning and its applications.

Dr. Aggarwal received Princeton University's Porter Ogden Jacobus Honorific Fellowship in 2009, the AT&T Vice President Excellence Award in 2012, the AT&T Senior Vice President Excellence Award in 2014, the 2017 Jack Neubauer Memorial Award recognizing the Best Systems Paper published in the IEEE Transactions on Vehicular Technology, the 2018 Infocom Workshop HotPOST Best Paper Award, and 2021 NeurIPS Cooperative AI Workshop Best Paper Award. He was on the Editorial Board of IEEE Transactions on Green Communications and Networking and the IEEE Transactions on Communications, and is currently on the Editorial Board of the IEEE/ACM Transactions on Networking, and the founding co-Editor-in-Chief of the ACM Journal on Transportation Systems.

Borja Peleato received the B.S. degrees in Telecommunications and Mathematics from the Universitat Politecnica de Catalunya, Barcelona, Spain, in 2007, and the M.S. and Ph.D. degrees in Electrical Engineering from Stanford University, Stanford, CA, USA, in 2009 and 2013, respectively. He was a visiting student at the Massachusetts Institute of Technology in 2006 and a Senior Flash Channel Architect with Proton Digital Systems in 2013. From 2014 to 2020, he was an Assistant Professor in the Electrical and Computer Engineering Department at Purdue University, West Lafayette, IN, USA. In 2020, he was awarded a CONEX-Marie Curie Fellowship and joined the Signal Theory and Communications group at the Universidad Carlos III de Madrid, Leganes, Spain, where he is currently a CAM Atraccion de Talento Fellow.

His research interests include wireless communications, information theory, convex optimization and nonvolatile storage.